Real-time machine learning: the next frontier?

The Applied AI Community

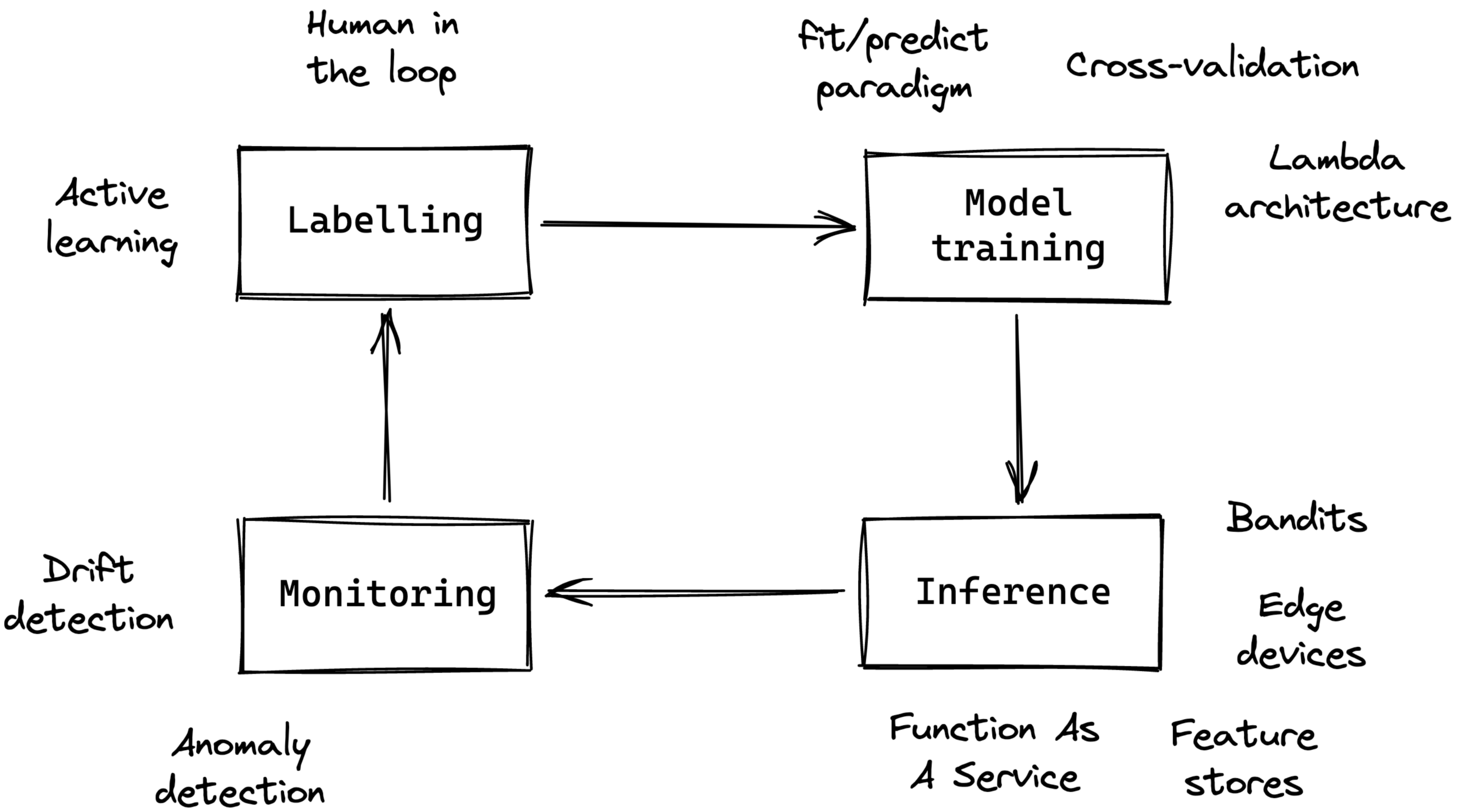
December 17th 2021

Max Halford

Hello, I’m Max! 👋

- 🤷‍♂️ I enjoy all aspects of data
- 😴 PhD in query optimisation
- 🥵 Kaggle competitions Master
- ❤️ I open source software
- ✍️ I like to blog
- 🐫 I’m a nomad

ML is maturing, design patterns are emerging

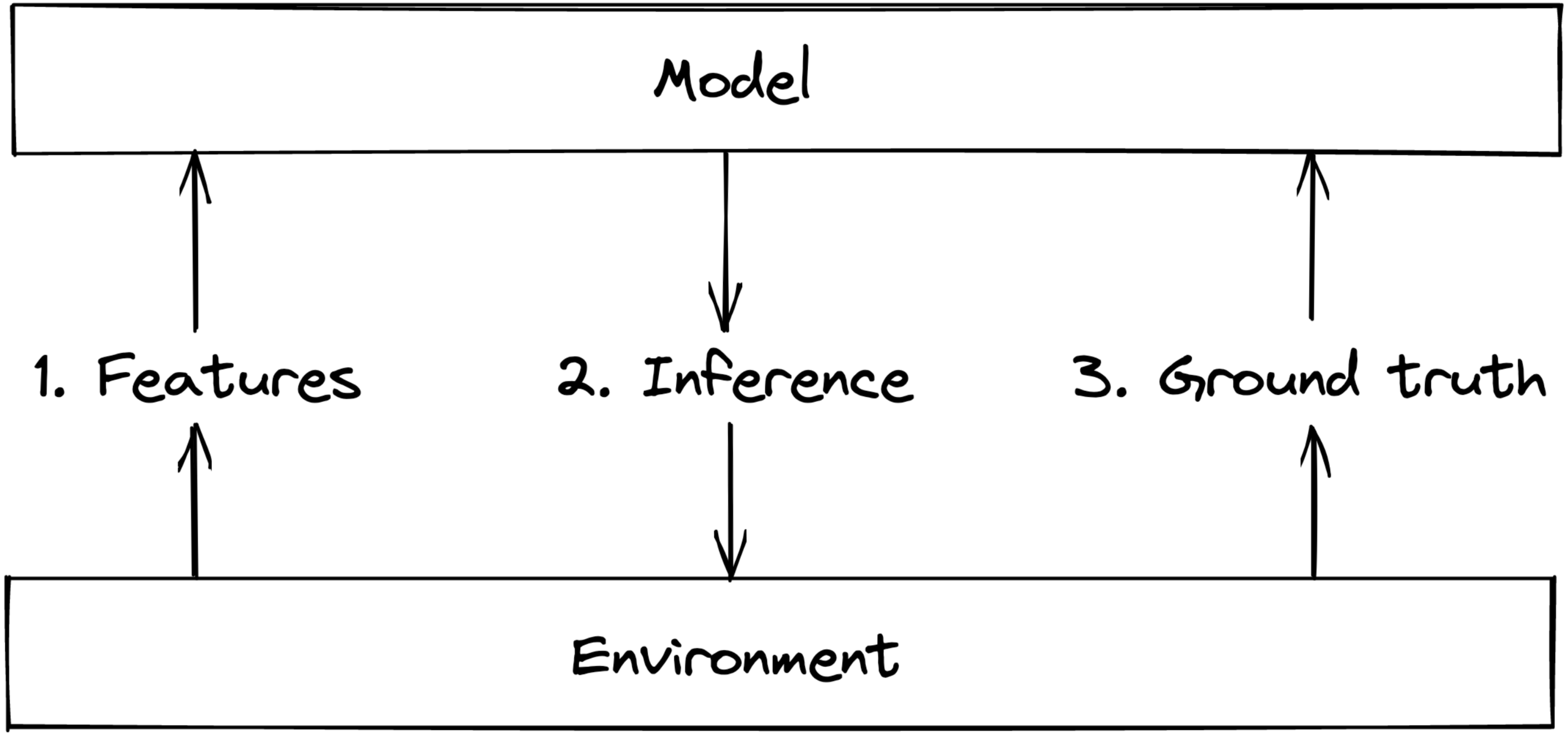


Batch learning is predominant

- Most ML models are batch models
- Batch models are trained on a static dataset
- Batch models have to be retrained from scratch
- Why is batch learning prevalent?

1. It’s what we’re used to, it’s **comfortable**
2. It’s **taught** at university
3. Huge **ecosystem**

Models are static, but the real world is dynamic



Real-time inference ≠ learning

- Predictions are traditionally done in batch
- Some companies are getting better at real-time inference
- You can already do interesting things here:
  - Shadow deployments
  - Canary deployments
  - Bandits
- But this is different to real-time learning
- Real-time learning is more difficult
- Do you need it?

A growing need for real-time learning

🍿 Netflix — update recommendations in-session

💱 Trading — learning as soon as possible gives an edge

🚕 Mobility — update routing from live traffic information

💰 Banking — fraud patterns constantly evolve

📡 Sensors — the definition of “normal” may change with time

🏴‍☠️ Cybersecurity — hackers adapt to defence strategies

🐤 Edge devices — can’t afford to store training data

“Real-time” is a weasel word

- There is no single definition
- Real-time means what you want it to mean
- Different applications will have different requirements
- You can fake it 🪄
- At the end of the day, what matters is the **business impact**

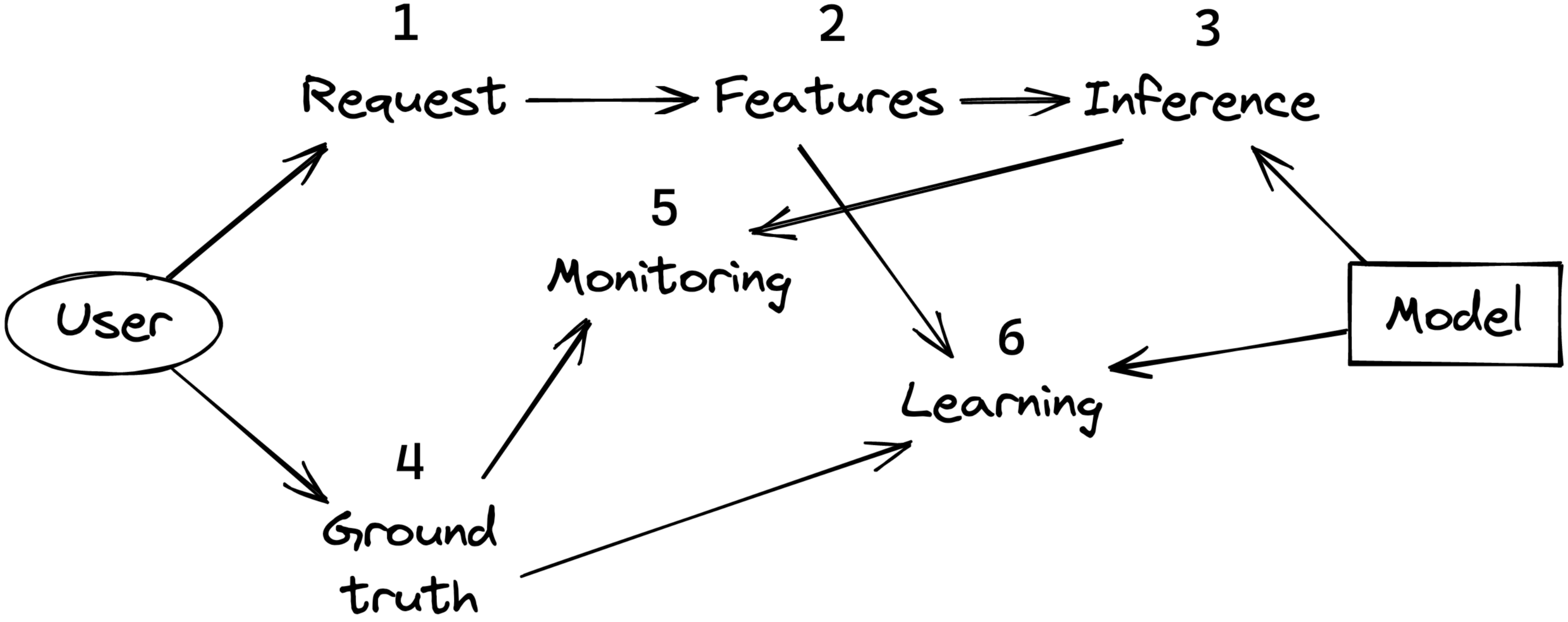
Batch retraining might be enough

- Usually, your system generates a stream of new training data
- Retrain your model periodically to cope with **concept drift**
- Maybe this works for you. If so, congrats 👏
- There are some downsides:
  - 🔋 It’s wasteful
  - ⏱️ A schedule needs to be decided
  - 🤔 **Online/batch parity** is not ensured

The alternative: online learning

- What if a model didn't have to be retrained from scratch?
- That's the mantra of online learning 🧘🏽
- An online model can learn from one sample at a time
- It keeps on learning without having to revisit past data

The online learning lifecycle



The benefits

🔋 It’s ecological because each sample is only seen twice

🧠 The model is always up-to-date

⏱️ No training schedule is necessary

🎯 Online/batch parity is ensured

🤝 Backtesting is reliable

🧙 It feels like magic when it’s running in production

Online/batch parity 🎯

- How do you ensure features are available at inference time?
- Leakage is always possible, even if you use a feature store
- In an online fashion, you predict and then you fit
- You train with features that were available during inference
- Online/batch parity is ensured 🤲

See *Building Faire's new marketplace infrastructure*

Progressive validation 📏

- Each data point `(x, y)` is used for inference and training
- First you predict, then you learn
- You can do an offline single pass over your dataset to
  A. train your model.
  B. obtain an out-of-fold score.

Delayed progressive validation 🔬

- You have control over the order in which the data is processed
- You can take into account the moment of arrival of x and y
- You can **reproduce offline what happened online**
- By doing this, you mimic production conditions
- This is **closer to reality** than cross-validation

See *The correct way to evaluate online machine learning models*

Stability-plasticity dilemma

- **Plasticity** — integrating new knowledge
- **Stability** — memorising previous knowledge
- Too much plasticity leads to **catastrophic forgetting**
- In an online setting, this **might not be a problem**
- Indeed, sometimes the goal is only to be good on recent data
- Continual/lifelong learning aims to address this dilemma

Why isn't online learning popular? 🤷‍♂️

- Batch learning is pervasive
- It requires a different mindset
- It requires a more mature data platform
- The ecosystem is not as flowering as with batch learning
- We're missing some success stories

River

🐍 Python library for online machine learning

🤝 Merger between creme and scikit-multiflow

😁 I've been working on this for roughly ~3 years

👷‍♀️ ~26,000 lines of code, ~2,450 unit tests

🚀 In production at a couple of companies

3 core developers from 🇫🇷 🇧🇷 🇳🇿

Beginner's example

```
>>> from river import compose
>>> from river import linear_model
>>> from river import metrics
>>> from river import preprocessing

>>> model = compose.Pipeline(
...     preprocessing.StandardScaler(),
...     linear_model.LogisticRegression()
... )

>>> metric = metrics.Accuracy()

>>> for x, y in dataset:
...     y_pred = model.predict_one(x)      # make a prediction
...     metric = metric.update(y, y_pred)  # update the metric
...     model = model.learn_one(x, y)      # make the model learn

>>> metric
Accuracy: 89.20%
```

Plain dictionaries are the building blocks

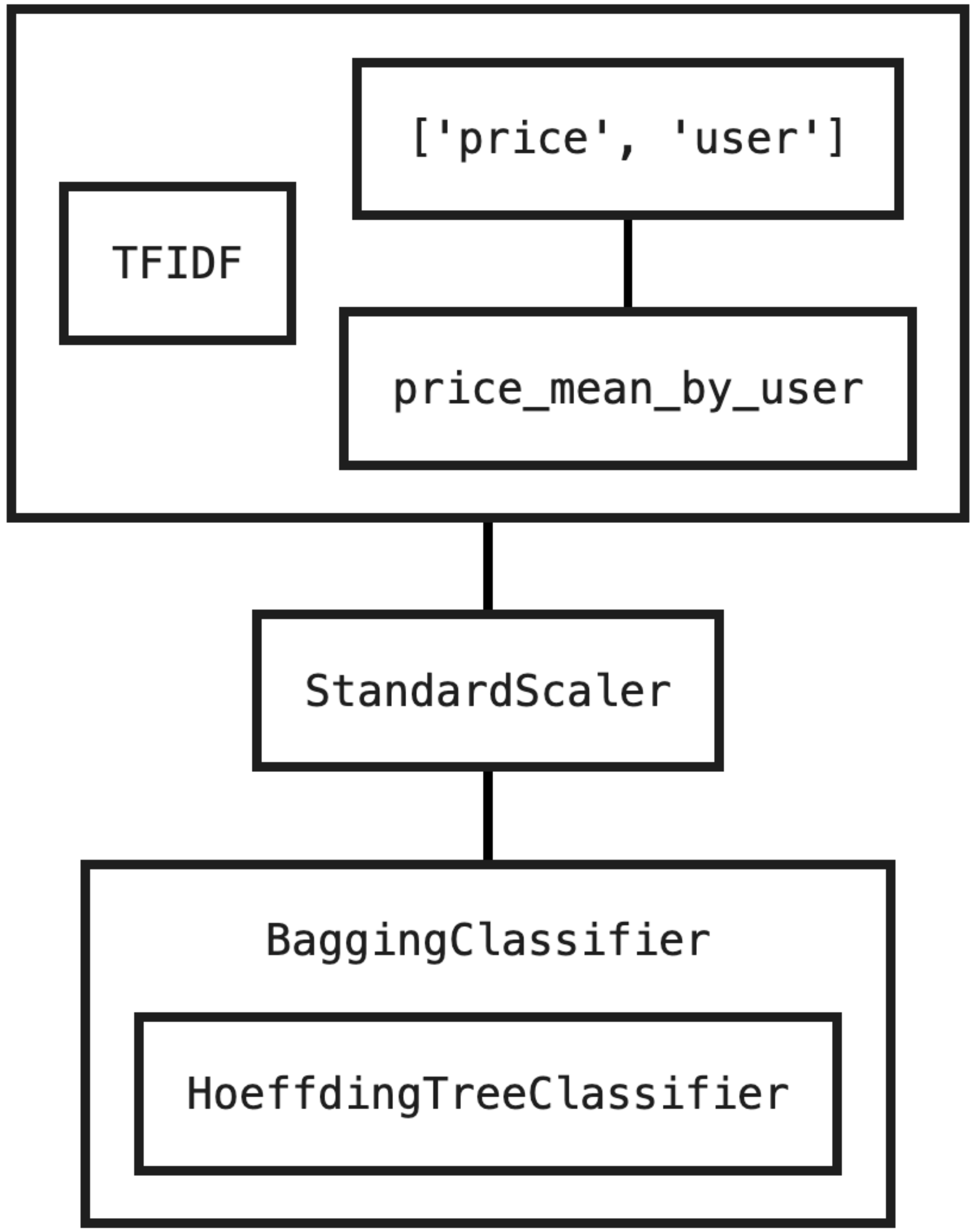
- Features are stored in dictionaries — called “dicts” in Python
- Allow naming features
- Are to lists what pandas DataFrames are to numpy arrays
- Naturally represent sparse data
- Native to Python → no overhead like numpy/pandas/torch
- JSON-friendly

Pipelines are first-class citizens

```
>>> from river import *

>>> model = compose.Pipeline(
...     (
...         feature_extraction.TFIDF(on='text') +
...         compose.Pipeline(
...             compose.Select('price', 'user'),
...             feature_extraction.Agg(
...                 on='price',
...                 by='user',
...                 how=stats.Mean()
...             )
...         )
...     ),
...     preprocessing.StandardScaler(),
...     ensemble.BaggingClassifier(
...         model=tree.HoeffdingTreeClassifier(),
...         n_models=10
...     )
... )

>>> model
```



It's a general-purpose library

Naive Bayes

Feature extraction

Preprocessing

Linear models

Factorization Machines

Nearest neighbors

Time series forecasting

Anomaly detection

Decision trees

Multi-output learning

Neural networks

Model selection

Imbalanced learning

Metrics

Streaming datasets

Clustering

Ensembling

Speed considerations

- Many libraries implement SGD, which allows comparing them
- River is optimised for pure online learning — single samples
- River shines when samples arrive one by one:
  - 10x faster than Vowpal Wabbit
  - 20x faster than scikit-learn
  - 50x faster than PyTorch
  - 180x faster than Tensorflow

What about processing huge datasets?

- Sometimes, size matters.
- Learning with one sample at a time is not efficient
- To go big, vectorisation is necessary
- Some River models can process mini-batches
- In conjunction with vaex, you can process millions of rows per second

☝️ **Pure online learning (i.e. individual samples) remains our main focus**

Is it being used?

- Yes, it is! 🎉
- We know a couple of companies who use it production
- We’ve heard rumours of it being used for prototypes
- The amount of traffic and discussions on GitHub is steady
- We don’t focus too much on fostering a widespread adoption
- Our goal is to satisfy the few teams who use River 🤝

Our roadmap

- Our roadmap is public, see here
- Tentative areas of focus for 2022:
  - Online learning on graphs
  - Recommendation systems
  - Reinforcement learning
  - Anomaly detection
  - Comprehensive benchmarks
  - Delightful documentation

Feel welcome to make suggestions 🤗

Thinking beyond River

- River is “just” a machine learning library
- It’s not an MLOps tool
- Deploying an online model requires some effort
- We see many people doing things differently 😵
- There is an opportunity to standardise streaming MLOps
- That’s my next endeavour 🤫

Takeaways

- Online machine learning isn’t a one-size-fits-all solution
- Batch learning is perfectly adequate for many problems
- Use the right tool for the job!
- Online learning needs more success stories to see adoption
- We’re friendly, so feel welcome to reach out 🤗

Further content

- Machine learning is going real-time — Chip Huyen
- maxhalford.github.io/links#talks — Yours truly
- One Pass ImageNet — DeepMind
- Machine learning with Flink in Weibo — Qian Yu
- Why TikTok made its user so obsessive? — Catherine Wang

Thank You

maxhalford.github.io

github.com/MaxHalford

maxhalford25@gmail.com